section 6

私のディジタル回路設計ノート

高精度のsin/cos発生回路

畔津明仁

前号に続いて，有効ビット数が多い(64ビット以上)演算回路の例を取り上げます．

三角関数(sin/cos)については，前々回(2000年4月号)でも説明しましたが，それは12～24ビット程度の精度を求める例でした．信号処理などでは，その程度の精度で十分なことがほとんどですが，一般の数値演算の場合，時として「遅くてもいいから精度が必要」なこともあります．

今回は，高精度を主眼とした関数発生について述べます．また，級数展開の手法や，浮動小数点における値域の問題にも触れます．

1．高精度のsin/cos

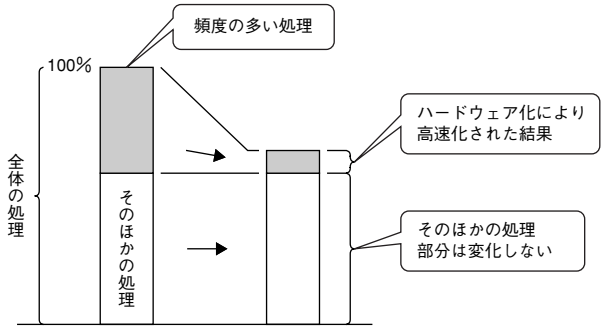
◆高精度の三角関数が必要なとき

狭義のsin(θ)，cos(θ)は，三角比と呼ばれるもので，主として平面三角法(**コラム**)に使用されます．“比”というのは，当然，辺長の比のことです．したがって，長さの精度と三角比(sin(θ),cos(θ))の精度は対応しています．たとえば，三角測量を考えると，距離の精度が10進6桁(2進だと20ビット)ならば，三角比もおおよそ同程度の精度があればいいことになります．

4月号で述べたsin，cosの発生についても，最終的には画像・音声あるいは信号処理(変調・復調など)の振幅に反映されるわけで，たとえば，目的とする“振幅”が16ビットなら，sin，cosのビット数も同程度で許されることになります．

このような考え方は，座標軸変換やフーリエ変換など，他のさまざまなアプリケーションにおいても成立します．つまり，sin(θ)やcos(θ)の精度というものは，“距離”，“振幅”，あるいは“動径”などの精度に対応していれば，ほとんどの用途では間に合います．アナログ値の測定を含む通常の信号処理において，その精度は24ビット程度以下です．

しかし，もちろん，例外は存在します．複雑な機械的リンク系の中間演算や，宇宙的規模の演算(動径がきわめて広範囲で，逆に角度は秒以下，つまり1回転に対して2^{-20}以下)などの場合です．また，たんに「途中結果が長語長であるため，関数も長語長でなければならない」といった要求も少なくありませんが…．



〔図1〕高速化の効果

◆ハードウェア化の意義

筆者の仕事として高精度(たとえば仮数部64ビット)の演算が要求された経緯は，前号で述べたとおりです．このような演算を高速化する手段として，一つは逆数(1/x)回路をハードウェア化しました(前号を参照)．

このほかに，どんな演算をハードウェア化すべきか？ が今月の問題です．

ここではCPUやDSPで行う数値計算を高速化するのが目的ですから，本来なら，アプリケーションごとの関数(マクロ)の呼び出し頻度を調べるのが役立つはずです．処理時間全体の中で，多くを占める演算をハードウェア化するのがもっとも有効です(**図1**)．

しかし，この方法で目標を絞るのは簡単ではありません．理由の一つは，アプリケーションごとに大幅な違いがあることです．第2は，コンパイラが吐き出すコードを調べても，すでに加算や乗算などの命令に変換されており，目的とした関数が何かわかりません．

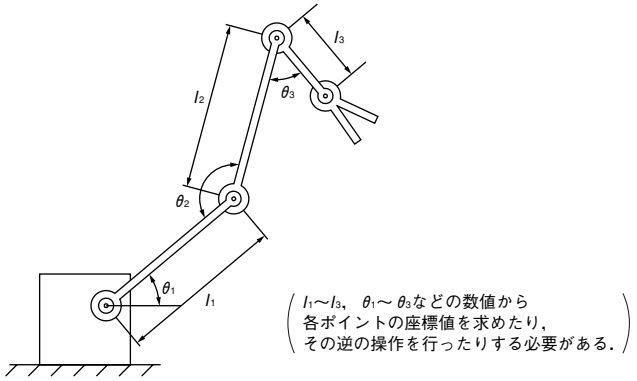
さらにもう一つの理由を上げておきましょう．それは，よほど特殊なアプリケーションでないかぎり，特定の演算だけが処理時間の10％以上を占めるようなことはないのです．つまり，特定の演算をハードウェア化しても，全体の性能向上は10％以下にすぎません．

今日，汎用CPUでは内部クロック周波数の上昇に主眼が置かれているのはご存知のとおりです．クロックの周波数を20％上げれば，性能は単純に20％向上します(もちろんメモリやI/Oの速度も連動する必要がありますが…)．苦労して

特定の演算をハードウェア化する意欲は薄れがちです．

本連載の4回目までに紹介したような信号処理用の演算回路では，必要なデータ･レートやレイテンシが明確に定まっており，それがソフトウェアで達成できない限り（データ･レートで10MHzを超えたら，ソフトウェアではまず対応できない．1 MHzでも無理な場合が多い）ハードウェア化せざるを得ません．

これに対して，もともとソフトウェアで行っていた演算の場合，上記のような理由から，へたにハードウェア化してもコストのむだになる可能性があります．設計者としてもっとも頭を使うべきなのは，この点です．



〔図2〕リンク機構

◆ なぜ，sin，cosか

悩みは少なくなかったのですが，筆者らはハードウェア化に取り組むことを決めました．次の問題は，どの演算をハードウェア化するかです．その一つとして選択したのは前号で述べた逆数（$1/x$）です．その理由は前号で触れたとおりです．

筆者らはさらに別種の関数のハードウェア化を考えました．候補としてあがったのは，対数（$\log_2(x)$），指数（2^x），2乗（x^2），3乗（x^3），平方根（$\sqrt{x}$），立方根（$\sqrt[3]{x}$），正接（$\tan(x)$），逆正接（$\arctan(x)$）などです．その中で最終的に正弦関数（$\sin(x)$，$\cos(x)$）を選びました．理由は次のとおりです．

⑴ アプリケーションの一つにリンク機構の計算があった

開発依頼者によると，アプリケーションの一つにリンク機構の計算があるという話でした（**図2**）．これには$\sin$，$\cos$が高頻度で登場します．

⑵ ソフトウェアとしては演算しにくい

ハードウェア化の効果を少しでも上げるには，ソフトウェアで演算しにくい（ステップ数が多い）ものを選ぶべきだと考えました．加減乗除はすでに除外してありますが，簡単に得られるx^2，x^3，$\sqrt{x}$，$\sqrt[3]{x}$などは対象外です．

コラム：平面三角比

“三角法”とは，三角形の各辺の長さ，各頂点の角度，面積などを求める操作です．その基本は，中学から高校で学んでいるはずです．

たとえば，三角形の面積について，小学校では，

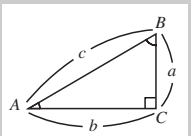
$s=ah/2$　（底辺×高さ÷2）

という公式を学びます．しかし，中学や高校を経ると，**表A**に示すようにさまざまな方法があることに気づくはずです．

さらに考えると，三角形の決定条件（これは三角形の合同条件に近い）が成立するのであれば，辺長，頂点の角度，面積のすべてが決定できなければならないはずです．

世の中の一般的な“三角測量”や“作図法”はすべて平面三角法を基本としています．そして，平面三角法の基礎は“三角比”つまり三角関数にあります．

図Aに示すのは，三角比の定義で，これを拡張したものが“三角関数”になります．一方，**表A**に示すのは三角法の一部です．もちろん実際には，自在なバリエーションが可能です．また，普通の“平面三角法”に相対するものとして，“斜軸（平面）三角法”や“球面（または曲面）三角法”がありますが，その基本はすべて平面三角法にあります．



$$\sin(A)=\frac{a}{c}$$
$$\cos(A)=\frac{b}{c}$$
$$\tan(A)=\frac{a}{b}$$
$$\sec(A)=\frac{c}{b}$$
$$\mathrm{cosec}(A)=\frac{c}{a}$$
$$\cot(A)=\frac{b}{a}$$

〔図A〕三角比（三角関数）の意味

〔表A〕平面三角法の一部

既知のもの	未知のものの算出方法		
	辺長	面積	頂角
辺長c 頂角$\angle A,\angle B,\angle C$	$a=\frac{c\times\sin(A)}{\sin(C)}$ $b=\frac{c\times\sin(B)}{\sin(C)}$	$s=\frac{1}{2}ab\times\sin(C)$ など多数	$\cos(A)=\frac{b^2+c^2-a^2}{2bc}$ など多数
辺長a,b 頂角$\angle C$	$c=\frac{a\times\sin(C)}{\sin(A)}$		
辺長a,b,c	―――	$s=\sqrt{p(p-a)(p-b)(p-c)}$ ただし$2p=a+b+c$ など多数	$\tan\left(\frac{A}{2}\right)=\sqrt{\frac{(p-b)(p-c)}{p(p-a)}}$ $\sin\left(\frac{A}{2}\right)=\sqrt{\frac{(p-b)(p-c)}{bc}}$ $\cos\left(\frac{A}{2}\right)=\sqrt{\frac{p(p-a)}{bc}}$ など多数